ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Laboratorio di
Sicurezza Informatica

Autorizzazione: modelli e implementazioni

Marco Prandini

Dipartimento di Informatica – Scienza e Ingegneria

Il processo di controllo dell'accesso

- Un soggetto autenticato deve essere autorizzato a svolgere operazioni sulle risorse del sistema
- Tre passaggi:
  - definire il modello del sistema controllato
    (limitatamente ai fattori critici per il controllo degli accessi)
  - definire la politica di accesso
    (le regole in base alle quali l'accesso è regolamentato)
  - attuare la politica
    (tramite opportuni meccanismi HW / SW)
- Come già detto è molto utile separare politiche e meccanismi
  - per confrontare diverse politiche senza essere sommersi dai dettagli di implementazione
  - per modellare i componenti al livello di astrazione più appropriato e identificare la serie minima di requisiti che qualsiasi sistema di controllo degli accessi dovrebbe rispettare
  - per progettare meccanismi come elementi costitutivi, utilizzabili per diversi tipi di politiche

Caratteristiche delle politiche

- Principio del privilegio minimo
  - qualsiasi accesso deve avvenire concedendo l'insieme di autorizzazioni più ristretto possibile
- Consistenza
  - deve esistere uno schema di risoluzione non ambiguo da impiegare quando è possibile applicare autorizzazioni diverse alla stessa richiesta di accesso
  - nessuna soluzione unica!
    - la regola più / meno specifica vince
    - default allow vs. default deny
    - vince la prima / ultima regola incontrata in ordine spazio / temporale
  - gerarchia degli autori delle regole
- Completezza e correttezza
  - qualsiasi richiesta di accesso deve ricevere risposta entro un limite di tempo predeterminato
  - ci deve essere una regola predefinita da applicare quando non è possibile trovare un'autorizzazione esplicita per una richiesta

Caratteristiche dei meccanismi

- Resistenza alle manomissioni
  - deve essere impossibile sabotare un meccanismo di controllo degli accessi senza che nessuno se ne accorga
- Principio di mediazione completa
  - ogni accesso alle risorse deve essere sottoposto al controllo e alla decisione del meccanismo
- Piccolo e autonomo
  - deve essere facile da testare e riparare
- Ragionevolmente economico
  - il suo costo non deve superare i danni causati da accessi non autorizzati

Parametri di decisione

- Identità del soggetto
  - ovvio
- Ruolo del soggetto
  - i soggetti possono assumere ruoli diversi
  - le decisioni di accesso vengono prese in base al ruolo attuale di un soggetto, indipendentemente dalla sua identità
- Modalità di accesso
  - la decisione è presa non solo in base all'identità / dal ruolo, ma anche al tipo di operazione che il soggetto vuole eseguire
- Vincoli spaziali e temporali
  - l'accesso può dipendere da dove si trova il soggetto e da quando viene effettuata la richiesta
- Storia delle attività svolte
  - possono essere imposti limiti alla quantità di attività di un soggetto e al tipo di utilizzo della risorsa

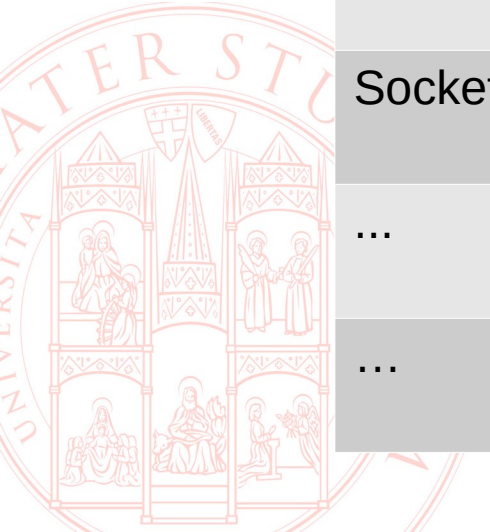
Modelli di controllo dell'accesso

- I due paradigmi fondamentali sono
  - DAC (Discretionary access control):
    - Ogni oggetto ha un proprietario
    - Il proprietario decide i permessi
  - MAC (Mandatory access control):
    - La proprietà di un oggetto non consente di modificarne i permessi
    - C'è una policy centralizzata decisa da un *security manager*
- Ci sono modelli più complessi
  - RBAC (Role-based access control):
    - I permessi sono assegnati ai *ruoli*
    - Utile se i soggetti possono assumere dinamicamente ruoli differenti a seconda del contesto (cosa devono fare, dove si trovano, in che tempi operano…)
  - e varianti...

Generalità sui meccanismi

- In principio, il controllo dell'accesso è decidere se un ***soggetto*** può eseguire una specifica ***operazione*** su di un ***oggetto***
- Il modo più banale per esprimere i *permessi* sarebbe una matrice completa
  - Migliaia di soggetti, milioni di oggetti!
  - La maggior parte delle "celle" è sempre al valore di default → potrebbe essere omessa

Subject / Object	User1	User2	Group3	...	...
File1	read	read write	--	...	...
Dir2	list	modify	--	...	...
Socket3	write	--	read	...	...
...	...	...	...	...	...
…	...	...	...	...	...



Implementazioni efficienti

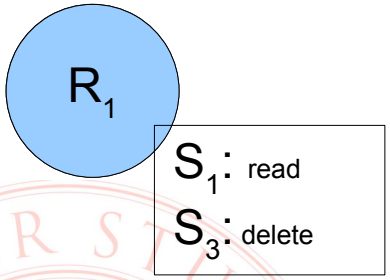
	R_1	R_2	R_3	R_4
S_1	read		modify	
S_2		read modify		
S_3	delete			
S_4			read	modify

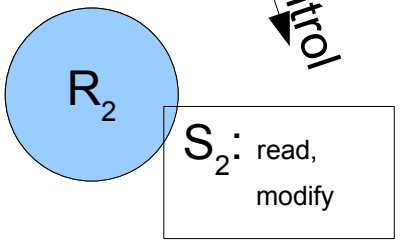
authorization table

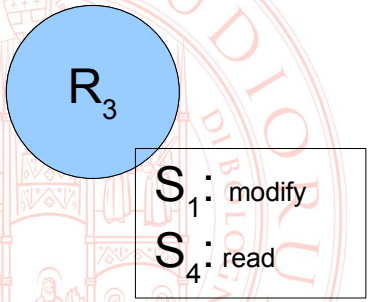
S	R	O
S_1	R_1	read
S_1	R_3	modify
S_2	R_2	read modify
S_3	R_1	delete
S_4	R_3	read
S_4	R_4	modify

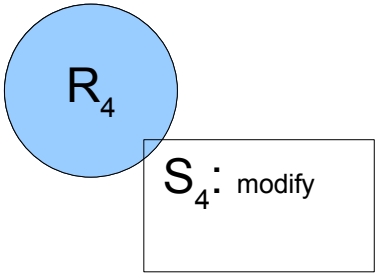
capabilities lists

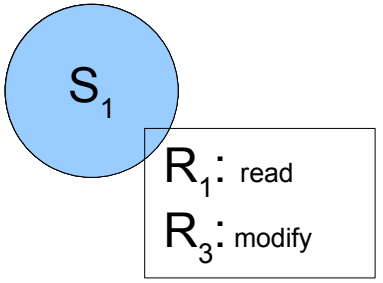
access control lists

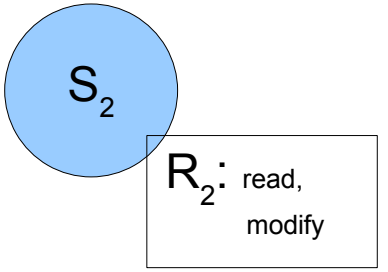


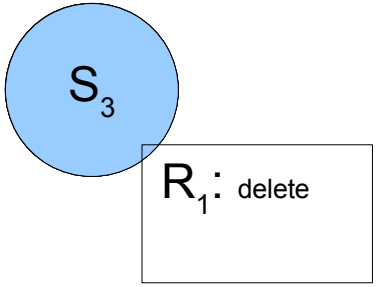


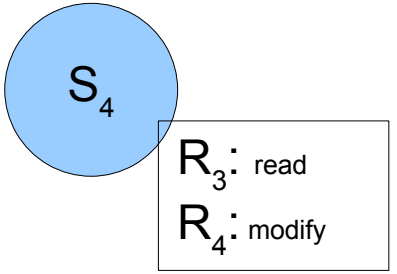












Implementazioni comuni

- Partizionare la matrice per soggetto: *capability lists*
  - Una lista associata a ogni soggetto del sistema
  - Contiene solo gli oggetti su cui il soggetto ha permessi ≠ default
- Partizionare la matrice per oggetto: *access control lists* (ACL)
  - Una lista associata a ogni oggetto del sistema
  - Contiene solo i soggetti che hanno permessi ≠ default sull'oggetto
  - Esplicitamente implementata da POSIX e MS Windows
  - Il filesystem Unix tradizionale ha negli inode una ACL "rigida", che elenca sempre e solo tre soggetti:
    - L'utente proprietario (U)
    - Il gruppo proprietario (G)
    - Il gruppo implicito che contiene tutti gli utenti ≠ U e $\notin$ G

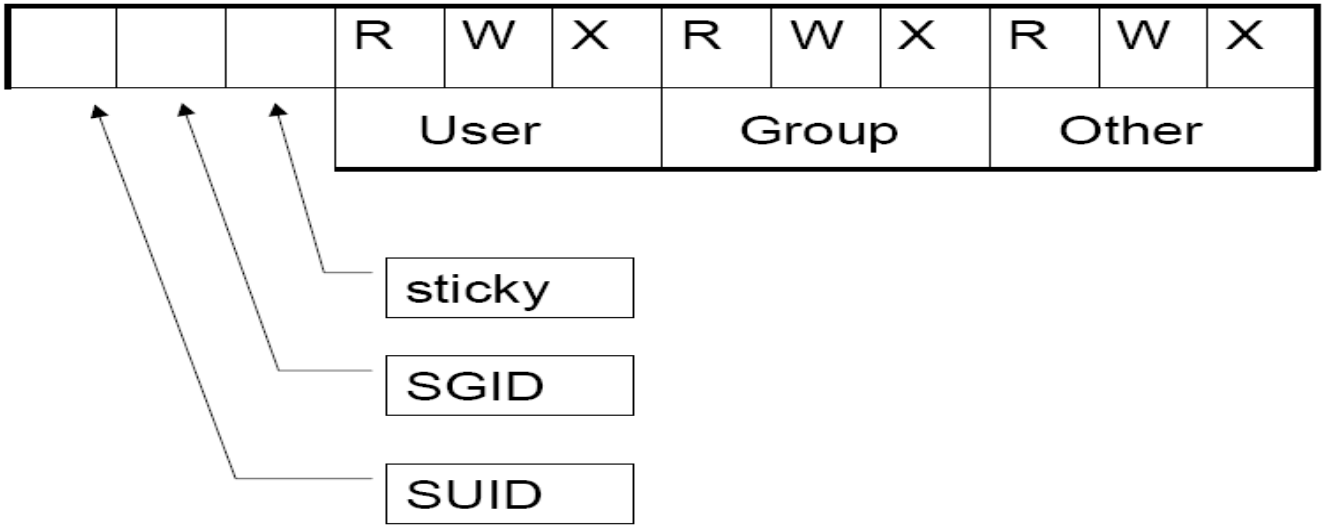
    e i relativi permessi

DAC nei sistemi Linux

- Gli utenti/gruppi possono essere creati con tool grafici o a riga di comando
  - `adduser`, `addgroup`
- Ogni utente DEVE appartenere almeno a un gruppo
  - Normalmente il sistema ne crea uno omonimo, con solo l'utente dentro
- Ogni utente può appartenere a un numero variabile di altri gruppi
- Gli account utente possono essere in uno stato *locked*, che impedisce di usarli per l'accesso interattivo, ma consente ai processi di girare con tale identità
  - Minimo privilegio!
- Il comando ***passwd*** si usa
  - Per cambiare le password (proprie, salvo root che può cambiarle a tutti)
  - Per settare l'account allo stato lock (`-l`) o unlock (`-u`)
    - Ovviamente solo root può farlo

Autorizzazioni su Unix Filesystem

- Ogni file (regolare, directory, link, socket, block/char special) è descritto da un i-node
- Un set di informazioni di autorizzazione, tra le altre cose, è memorizzato nell'i-node
  - (esattamente un) utente proprietario del file
  - (esattamente un) gruppo proprietario del file
  - Un set di 12 bit che rappresentano permessi standard e speciali



Significato dei bit di autorizzazione

- Leggermente diverso tra file e directory, ma in gran parte deducibile ricordando che
  - Una directory è semplicemente un file
  - Il contenuto di tale file è un database di coppie (nome, i-node)

R = read (lettura del contenuto)

Lettura di un file

Elenco dei file nella directory

W = write (modifica del contenuto)

Scrittura dentro un file

Aggiunta/cancellazione/rinomina di file in una directory

X = execute

Esegui il file come programma

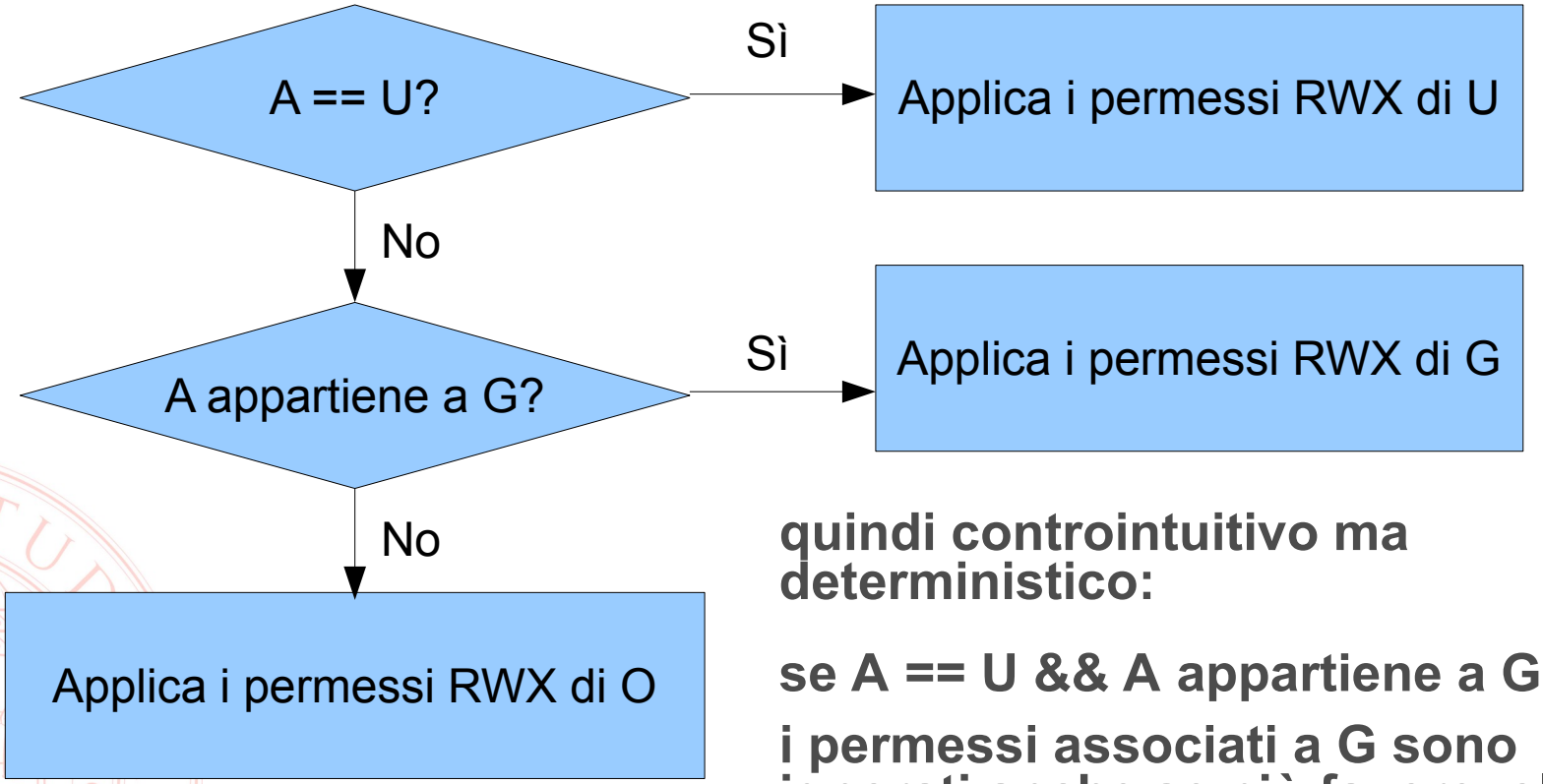
Esegui il lookup dell'i-node nella

NOTA che il permesso 'W' in una directory consente a un utente di cancellare file sul contenuto dei quali non ha alcun diritto

NOTA: l'accesso a un file richiede il lookup di tutti gli i-node corrispondenti ai nomi delle directory nel path → serve il permesso 'X' per ognuna, mentre 'R' non è necessario

Composizione dei permessi

- Quando un utente “A” vuole eseguire un’operazione su di un file, il sistema operativo controlla i permessi secondo questo schema:



quindi controintuitivo ma deterministico:

se A == U && A appartiene a G

i permessi associati a G sono ignorati anche se più favorevoli

Controllo dei permessi predefiniti

- Servono automatismi per assegnare i permessi alla creazione
- Ownership
  - l'utente creatore è assegnato come proprietario del file
  - Il gruppo attivo dell'utente creatore è assegnato come gruppo proprietario
    - Default = gruppo predefinito, da */etc/passwd*
    - L'utente lo può cambiare a mano nella sessione con **newgrp**
    - Può cambiare automaticamente nelle directoy con SGID settato
- Permessi = "tutti quelli sensati" tolta la umask
  - "tutti quelli sensati"
    - `rw-rw-rw-` (666) per i file, l'eseguibilità è un'eccezione
    - `rwxrwxrwx` (777) per le directory, la possibilità di entrarci è la regola
    - la *umask* quindi può essere unica: una maschera che toglie i permessi da non concedere
  - poiché in Linux il gruppo di default group di un utente contiene solo l'utente stesso, una umask sensata è 006 (toglie agli "other" lettura e scrittura)
    - È un settaggio utile per collaborare, crea file manipolabili da tutti i membri del gruppo, a patto che questo sia settato correttamente
  - col comando **umask** si può interrogare e settare interattivamente, per rendere persistente la scelta si usano i file di configurazione della shell

Bit speciali / per i file

I tre bit più significativi della dozzina (11, 10, 9) configurano comportamenti speciali legati all'utente proprietario, al gruppo proprietario, e ad altri rispettivamente

- BIT 11 – SUID (Set User ID)
  - Se settato a 1 su di un programma (file eseguibile) fa sì che al lancio il sistema operativo generi un processo che esegue con l'identità dell'utente proprietario del file, invece che quella dell'utente che lo lancia
- BIT 10 – SGID (Set Group ID)
  - Come SUID, ma agisce sull'identità di gruppo del processo, prendendo quella del gruppo proprietario del file
- BIT 9 – STICKY
  - OBSOLETO, suggerisce al S.O. di tenere in cache una copia del programma

Permessi delicati da tenere sotto controllo

- SUID e SGID sono un modo efficace di implementare interfacce per utenti standard verso processi privilegiati
  - Cambio password: guardare /usr/bin/passwd
  - Pianificazione di attività: guardare /usr/bin/crontab e /var/spool/cron/
  - etc.
- I programmi con questi permessi vanno sorvegliati, perché chiunque li lanci acquisisce temporaneamente privilegi elevati
  - Pochi programmi e molto vincolati
  - Rischi: bug di questi programmi che porti a eseguire operazioni arbitrarie invece di quelle progettate, programmi diversi a cui sono dati per errore questi privilegi
- Usare find per trovarli
  - `find / -type f -perm +6000`
- Altre ricerche interessanti per la sicurezza
  - file world-writable (`-perm +2`)
  - file senza proprietario, rimasti da account cancellati (`-nouser`)

Bit speciali / per le directory

- Bit 11 per le directory non viene usato
- Bit 10 – SGID
  - Precondizioni
    - un utente appartiene (anche) al gruppo proprietario della directory
    - il bit SGID è impostato sulla directory
  - Effetto:
    - l'utente assume come gruppo attivo il gruppo proprietario della directory
    - I file creati nella directory hanno quello come gruppo proprietario
  - Vantaggi (mantenendo umask 0006)
    - nelle aree collaborative il file sono automaticamente resi leggibili e scrivibili da tutti i membri del gruppo
    - nelle aree personali i file sono comunque privati perché proprietà del gruppo principale dell'utente, che contiene solo l'utente medesimo
- Bit 9 – Temp
  - Le "directory temporanee" cioè quelle world-writable predisposte perché le applicazioni dispongano di luoghi noti dove scrivere, hanno un problema: chiunque può cancellare ogni file
  - Questo bit settato a 1 impone che nella directory i file siano cancellabili solo dai rispettivi proprietari

Attributi

- Gli attributi sono primariamente utili per il fs tuning
  - compressed (c), no dump (d), extent format (e), data journalling (j), no tail-merg-ing (t), undeletable (u), no atime updates (A), synchronous directory updates (D), synchronous updates (S), and top of directory hierarchy (T)
- Alcuni sono rilevanti per la sicurezza
  - append only (a) – utile per impedire il taglio dei logfile
  - immutable (I) – vieta cancellazione, creazione di link verso il file, rinomina e scrittura, utile per i file di sistema
  - secure deletion (s) – sovrascrive con zeri i blocchi dei file cancellati (sicurezza molto limitata ma valida contro strumenti in linea)
- Tools
  - **chattr** per modificarli
  - **lsattr** per visualizzarli

POSIX Access Control Lists

- Le ACL estendono la flessibilità di autorizzazione
- Vantaggi:
  - Specificare una lista arbitraria di utenti e gruppi coi relativi permessi (comunque scelti tra rwx) in aggiunta agli owner
  - Ereditare la maschera di creazione dalla directory
  - Limitare tutti i permessi simultaneamente (esempio mask sotto)
- Esempio:

```
user::rw-
user:lisa:rw-            #effective:r--
group::r--
group:toolies:rw-        #effective:r--
mask::r--
other::r--
```

- Strumenti:
  - `setfacl` per impostare, `getfacl` per visualizzare le ACL (ls -l mostr un '+' dopo i permessi se ACL è presente per un file)
  - `man acl`

Il super-utente nei modelli DAC

- Esiste tipicamente un utente con privilegi illimitati, che può scavalcare i meccanismi di controllo dell'accesso
  - *root* in Unix
  - *administrator* in Windows
- L'account va difeso contro ingressi abusivi ma va anche minimizzata la probabilità di fare errori
  - Usare un account non-privilegiato, basta per il 99% del tempo
  - Disabilitare l'accesso diretto da GUI e console
  - Ottenere temporaneamente i diritti di super-utente solo per eseguire i task di amministrazione
    - *sudo* in Linux
    - "*esegui come amministratore*" in Windows

Capabilities in Linux

(da non confondere con le capability list tipiche dei modelli MAC)

- I poteri di *root* non sono "monolitici"
- Ci sono ben 41 diverse *capability* (al kernel 5.9)
  - rappresentano autorizzazioni normalmente negate agli utenti standard
  - riguardano molteplici aspetti di controllo delle risorse di calcolo e dei processi e dell'accesso alla rete
  - una nello specifico è CAP_DAC_OVERRIDE: la possibilità di ignorare i permessi sul filesystem
- È possibile assegnare specifiche capability a processi lanciati da utenti standard
  - autorizzazione a svolgere azioni privilegiate senza accesso a root
  - implementazione del principio di minimo privilegio
- `man (7) capabilities (8) getcap (8) setcap`

DAC nei sistemi Microsoft

- Utenti e loro proprietà
- Tipi di gruppi
- Estensione dei gruppi sui domini
- Gruppi predefiniti
- Generalità NTFS
- Implementare la sicurezza di NTFS
- Implementare la condivisione di risorse
- Permessi locali e permessi sulle condivisioni NTFS

Premessa: i domini

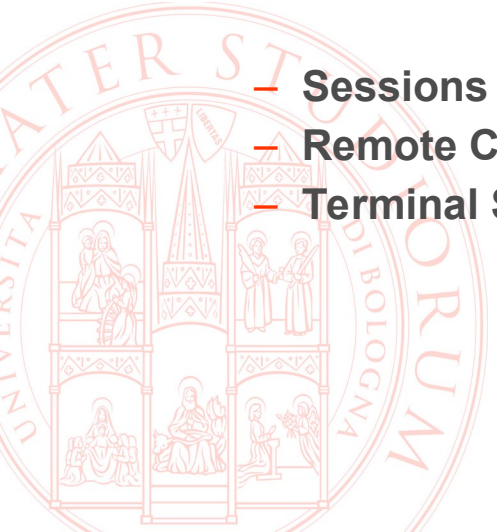
- Nell'uso più comune, i sistemi Microsoft sono raggruppati in un *dominio*: un insieme di computer, comunicanti tra loro e che condividono un directory database comune
- Nella directory sono memorizzati vari tipi di oggetti
  - I computer che fanno parte del dominio
  - Le risorse condivise dai computer (cartelle, stampanti)
  - Gli utenti validi sul dominio
  - I gruppi di utenti
  - I raggruppamenti di altre entità
  - ...

Utenti

- Local user accounts
  - ristretti al sistema su cui sono creati
  - possono avere moderati permessi amministrativi (che non si estendono alla possibilità di accedere ai dati di altri utenti) --> Power Users Group
- Domain user accounts
  - appartiene ad un dominio
  - profilo memorizzato in AD
  - può accedere a risorse non locali, limitatamente ai privilegi che gli sono concessi
    - del proprio dominio
    - dei domini trusted

Proprietà dell'utente

- Sono moltissime,accessibili dai *tab* del wizard qui elencati:
  - Member Of — The user's defined group membership
  - Dial-in — Remote access and callback options
  - General — User's first name, last name, display name description, office location, telephone, e-mail, and Web pages
  - Address — User's post office mailing address
  - Account — Logon name, domain, logon hours, logon to server name, account options, and account expiration date
  - Profile — User profile path, profile script, home directory path and server, and shared document folder location
  - Telephones/Notes — Home, pager, and mobile phone numbers and comments on where to contact user
  - Organization — Job title, company, department, manager, and people who report to user Environment Applications to run from Terminal server client
  - Sessions — Timeouts for Terminal Services
  - Remote Control — Permissions for monitoring Terminal Service sessions
  - Terminal Service Profile — Location for Terminal Service home directory



Gruppi

- Ogni oggetto di AD può essere membro di uno o più gruppi (di tipo e scope appropriato)
- Distribution Groups
  - possono essere usati da qualsiasi applicazione abbia bisogno di una lista di utenti
  - il sistema operativo non li utilizza
    - non appesantiscono il logon ticket dell'utente
- Security Groups
  - come i DG, ma possono essere soggetti nelle regole che controllano l'accesso alle risorse del sistema
- In Windows 2003 funzionante in Native Mode è possibile la conversione da un tipo all'altro

Group scopes

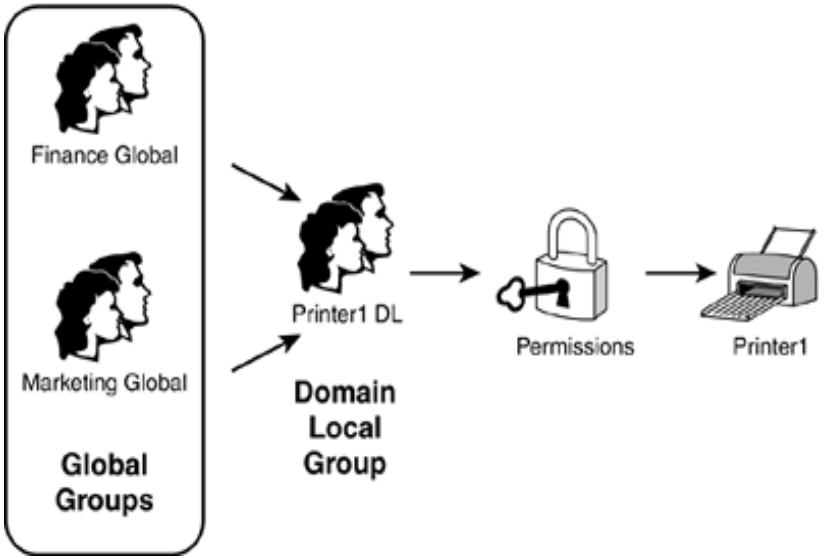
- Sia per i Distribution Group che per i Security Group vale il concetto di *scope* (estensione), che definisce
  - oggetti di quali domini possono far parte del gruppo
  - in quali domini può essere usato un gruppo per definire regole d'accesso
- La prima suddivisione è tra
  - Machine Local (locali ad una singola macchina)
  - Gruppi validi nel dominio
    - Domain Local
    - Global
    - Universal
- Nesting: è possibile *solo in native mode* rendere gruppi membri di altri gruppi

Group scopes

	Può contenere	Può essere membro di	Gli possono essere assegnati permessi su
Domain Local Group (DLG)	Utenti, GG, UG, computer di qualsiasi dominio, DLG dello stesso dominio	Altri DLG dello stesso dominio	Risorse dello stesso dominio
Global Group (GG)	Utenti ed altri GG dello stesso dominio	Qualsiasi DLG e UG, GG dello stesso dominio	Risorse di qualsiasi dominio
Universal Group (UG)	Utenti, GG, UG di qualsiasi dominio	DLG e UG di qualsiasi dominio	Risorse di qualsiasi dominio

Utilizzo tipico e consigliato

- Sebbene sia possibile assegnare diritti su risorse direttamente a UG e GG, la struttura consigliata è (caso più semplice):
  - individuare in ogni dominio utenti con esigenze analoghe e metterli in un GG
  - rendere i GG membro degli opportuni DLG
  - assegnare i permessi d'uso delle risorse ai DLG

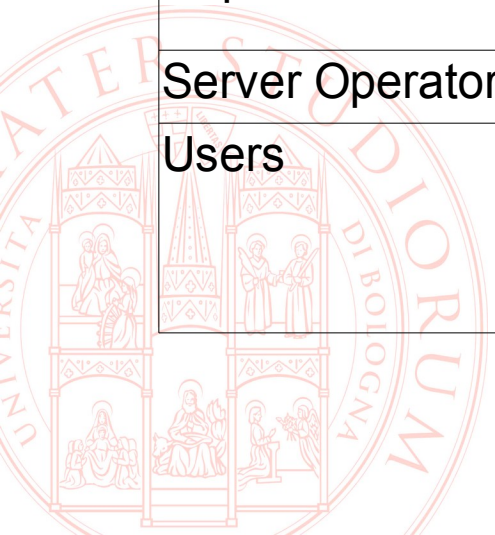


Utilizzo tipico e consigliato

- In realtà molto ampie è possibile sfruttare gli UG per aggiungere un livello di nesting che consenta all'*enterprise administrator* di raggruppare i GG
  - individuare in ogni dominio utenti con esigenze analoghe e metterli in un GG
    - in questo modo si delega al *domain administrator* che conosce bene la propria realtà il compito di popolare i GG
  - raggruppare i GG omologhi in un UG
    - in questo modo si evita che alla riorganizzazione dei domini, o in generale alla comparsa/scomparsa di GG, i singoli amministratori delle risorse debbano agire sui DLG, ripopolandoli di conseguenza. Sarà l'*enterprise administrator* a sapere quali GG è opportuno assegnare agli UG, mentre questi ultimi saranno creati o distrutti solo in casi eccezionali.
  - rendere l'UG membro degli opportuni DLG
  - assegnare i permessi d'uso delle risorse ai DLG
    - gli amminstratori delle singole risorse possono scegliere i soggetti (GG e UG) preconfigurati ai passi precedenti come membri di DLG, anzichè come soggetti cui attribuire direttamente permessi, in modo che l'aggiunta o la rimozione di un UG/GG da un DLG si applichi automaticamente a tutte le risorse su cui tale DLG può operare

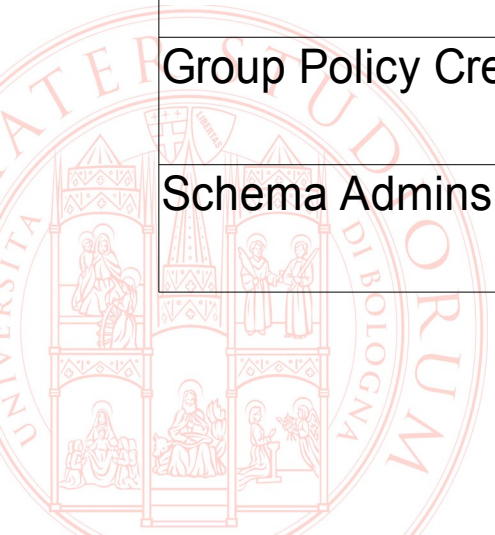
Gruppi predefiniti – domain local

Gruppo	Caratteristiche
Administrators	Controllo completo della macchina locale con tutti i provilegi; membri di default comprendono i Domain Admins, gli Enterprise Admins, e l'account Administrator.
Account Operators	Amministrazione degli utenti del dominio.
Backup Operators	Back up e restore dei file sulla macchina locale indipendentemente dai permessi ad essi associati; log on e shut down. Le Group policies possono limitare questi privilegi di default.
Guests	Logon/shutdown limitato sulla macchina locale.
Print Operators	Amministrazione delle stampanti locali.
Replicator	Gestione delle funzioni e dei servizi di replica di Active Directory.
Server Operators	Amministrazione del sistema locale.
Users	Esecuzione di applicazioni, accesso alle stampanti, logon/shutdown/locking, creazione e modifica fi gruppi locali; tutti gli utenti del dominio sono membri di default.



Gruppi predefiniti – global

Gruppo	Caratteristiche
Domain Admins	Privilegi di amministrazione su tutti i sistemi appartenenti al dominio
Domain Computers	Tutti i computer del dominio
Domain Controllers	Tutti i domain controller
Domain Guests	Appartiene al DLG “Guest”
Domain Users	Appartiene al DLG “Users”
Enterprise Admins	Appartiene al gruppo “Domain Admins” di ciascun dominio, concedendo quindi i privilegi di amministrazione a livello di foresta.
Group Policy Creators Owners	Ai membri è consentito modificare le group policy
Schema Admins	Ai membri è consentito modificare lo schema di Active Directory



Group Policy

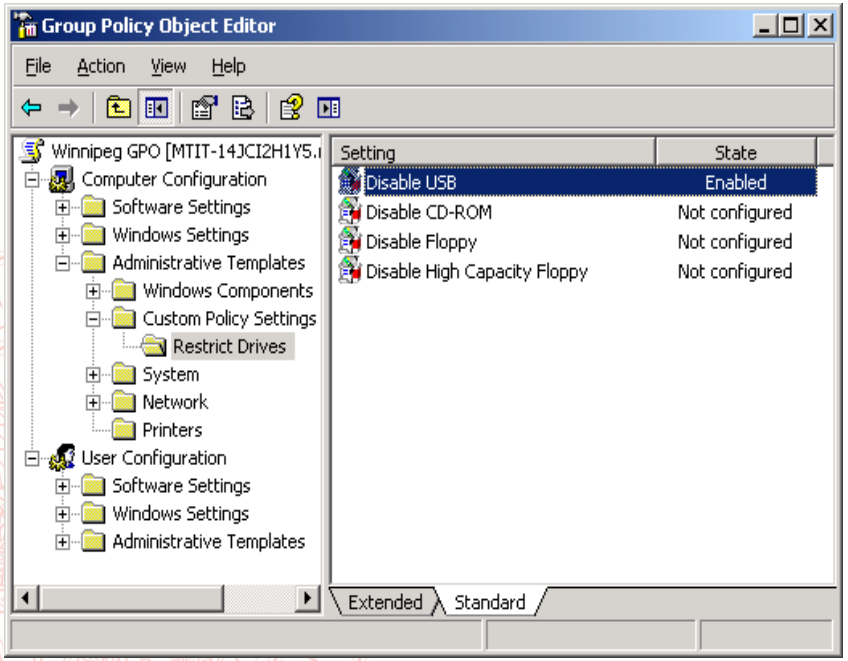
- Group Policy fornisce un quadro di riferimento per controllare l'ambiente di utenti e computer, cioè per assegnare quel tipo di privilegi o restrizioni che non sono legati a risorse fisiche ovvie quali file, cartelle, ecc.
- Le regole vengono definite in un Group Policy Object, che può essere collegato a qualsiasi contenitore di oggetti (una OU, un Site, un Domain) per applicarle a tutti gli oggetti in esso contenuti.
- Ogni GPO contiene due sezioni distinte
  - impostazioni per gli utenti (user settings)
  - impostazioni per i computer (computer settings)
- In ciascuna delle due sezioni le impostazioni sono ulteriormente classificate in:
  - impostazioni software (software settings)
  - impostazioni di Windows (Windows settings)
  - modelli per l'amministrazione (administrative templates)

Group Policy - impostazioni

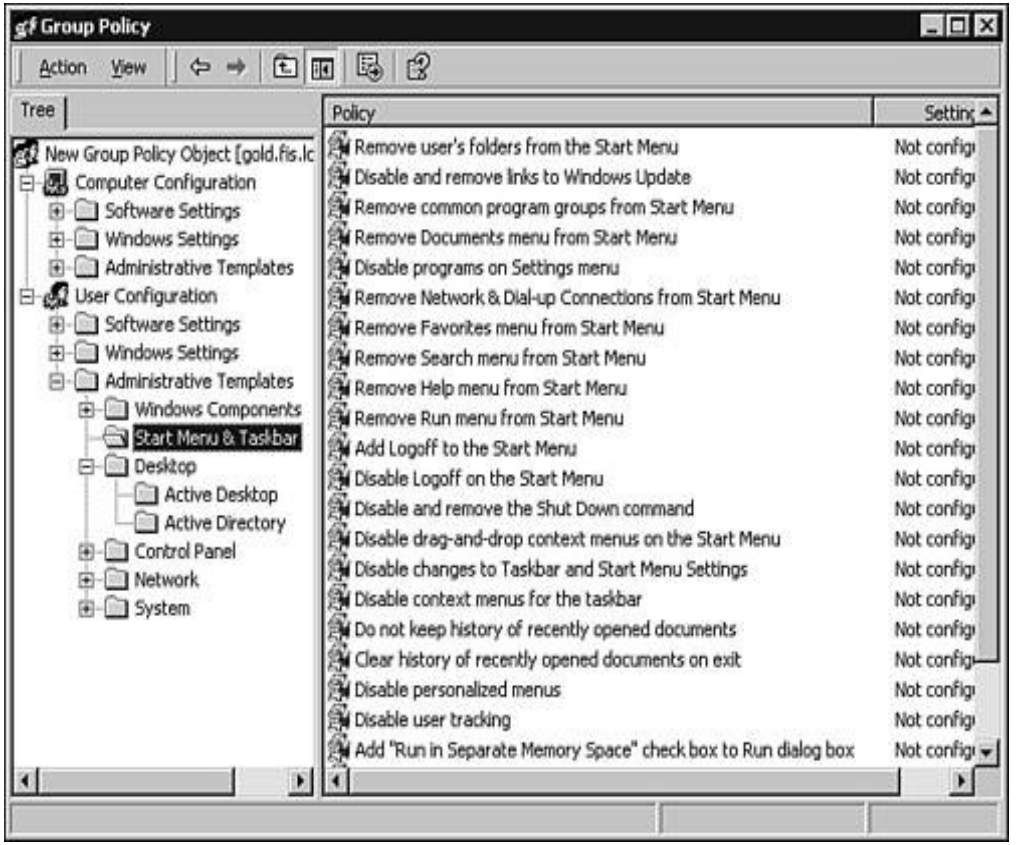
Categoria	Finalità	Disponibile per computer?	Disponibile per utenti?
Software settings	Installare, aggiornare, rimuovere applicazioni	Sì	Sì
Windows settings	Definire scripts ed impostazioni di sicurezza (vedi colonne a fianco)	Start-up e shutdown scripts, numerose impostazioni di sicurezza	Logon e logoff scripts, alcune impostazioni di sicurezza, impostazioni di Internet Explorer, folder redirection
Administrative templates	Definire in modo centralizzato le impostazioni del registro di sistema	Sì	Sì

Group Policy - esempi

Limitare l'uso di una intera categoria di dispositivi, come le porte USB

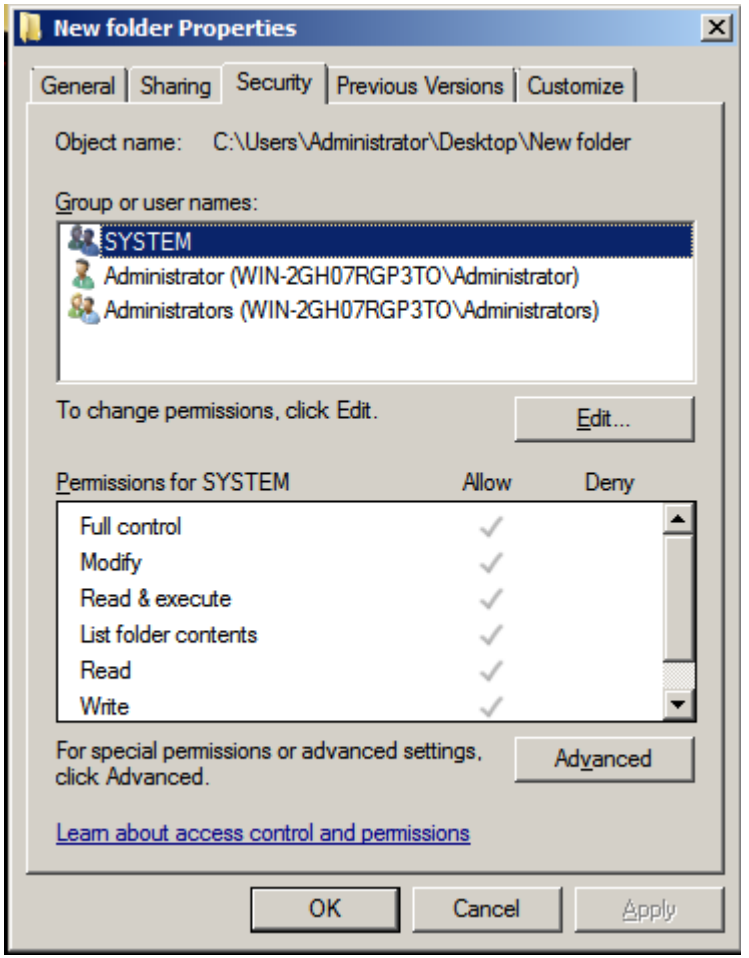


Configurare il contenuto del desktop o del menu avvio

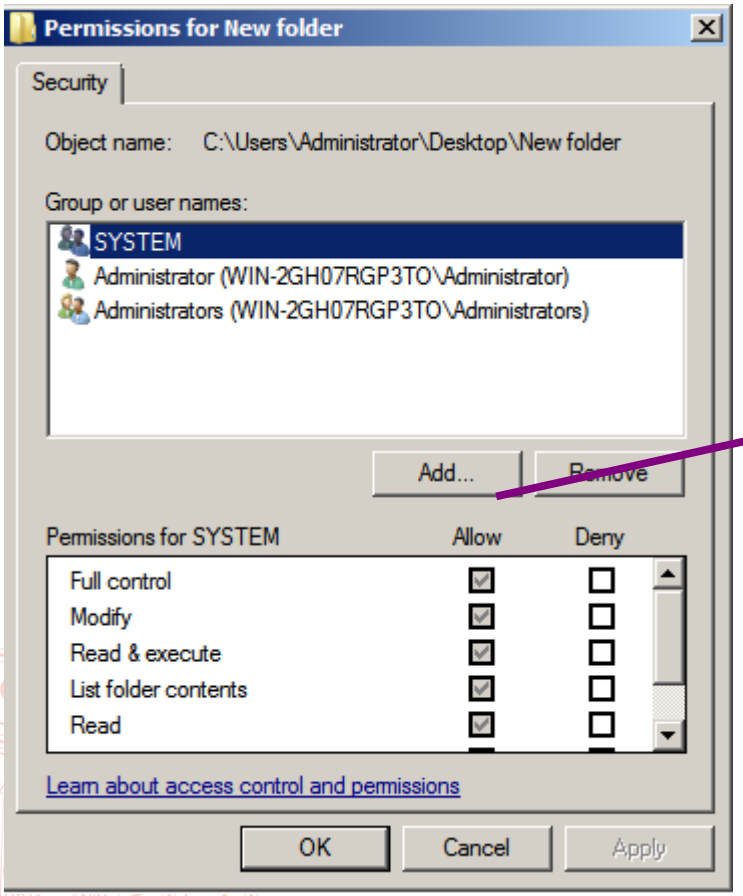


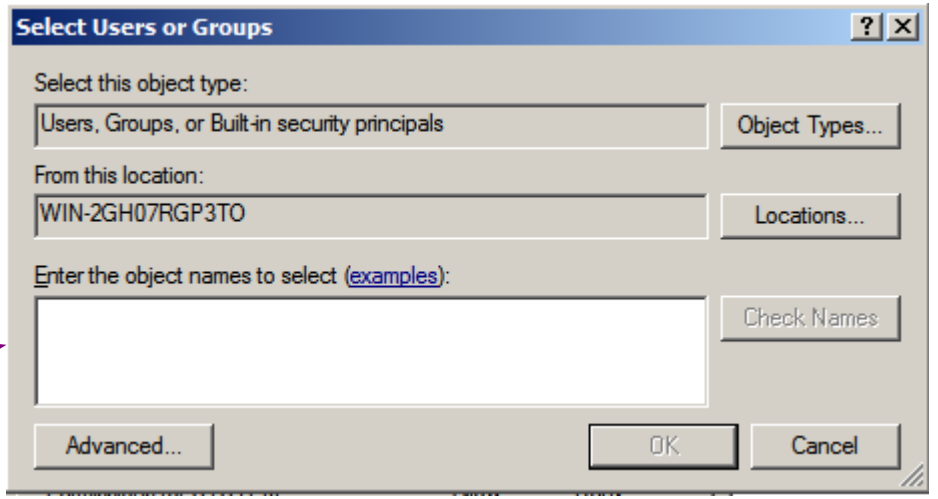
Controllo dell'accesso

- Le autorizzazioni sono assegnate sotto forma di ACL: ad ogni risorsa è associata una lista di soggetti (utenti o gruppi) e dei relativi permessi che essi detengono sulla risorsa
- Le ACL sono disponibili
  - su partizioni NTFS (non FAT)
  - sulle condivisioni di risorse in rete
- Per modificare le ACL è necessario
  - o detenere l'Ownership
  - o che nell'ACL medesima siano assegnati i permessi 'Full Control' o 'Change Permissions'



Esempio di modifica delle ACL





Aggiunta di soggetti alla lista collegata ad una risorsa

(da "Edit" della finestra precedente)

Ownership ed autorizzazioni

- Ownership
  - L'Owner di files e directories ha il pieno controllo (Full Control)
  - Administrator può sempre prendere l'ownership
  - L'Owner può assegnare le permissions per prendere l'Ownership
  - Nota: gli utenti che creano un file o una directory ne detengono l' Ownership
- Autorizzazioni NTFS predefinite
  - Ad Everyone viene assegnato automaticamente Full Control
  - I nuovi file ereditano le autorizzazioni della cartella in cui vengono creati (questo vale anche per i files che vengono copiati in un direttorio)

Accesso ed auditing

- Le ACL per il controllo dell'accesso fanno sì che, ad ogni tentativo di utilizzo di una risorsa, il sistema risponda autorizzando o negando l'operazione
- Ad ogni risorsa è inoltre associata una SACL utilizzata per l'auditing, che si presenta come una normale ACL, ma permette di tracciare gli esiti dei tentativi di utilizzo
- Le regole nella SACL possono essere impostate in modo che
  - quando un determinato soggetto tenta un'operazione e, grazie alla configurazione della ACL standard, *riesce*, questo evento sia registrato
  - quando un determinato soggetto tenta un'operazione e, grazie alla configurazione della ACL standard, *viene bloccato*, questo evento sia registrato

Autorizzazioni standard e speciali

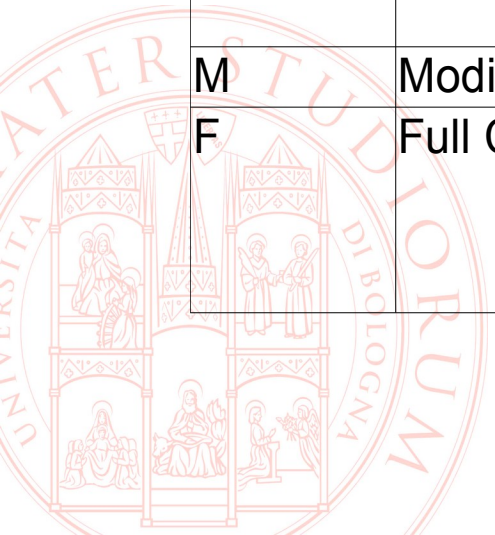
- L'obiettivo del sistema di controllo delle autorizzazioni è duplice
  - elevata precisione nel controllo dell'accesso
    - in termini di tipo di azioni da concedere/negare
    - in termini di gestione delle complesse relazioni tra utenti e gruppi che possono essere titolari delle autorizzazioni
  - facilità d'uso
    - il sistema è Discretionary Access Control (DAC), quindi consente ad ogni utente anche non tecnico di manipolare le autorizzazioni sulle proprie risorse
    - anche l'utente più esperto per il 90% del tempo fa cose semplici

Autorizzazioni standard e speciali (cont.)

- La soluzione ai due problemi è realizzata con un sistema che prevede tre strati di interfaccia utente per “svelare” all'occorrenza i dettagli che servono:
  - a basso livello il sistema supporta
    - molte autorizzazioni (*autorizzazioni speciali*) --> possibilità di controllo fine sui permessi
    - con una logica a tre valori (*allow, deny, not set*) --> possibilità di definire regole di interazione quando diverse ACL vengono combinate
  - le autorizzazioni speciali sono aggregate in un set più ridotto di *autorizzazioni standard*
  - le autorizzazioni standard possono essere visualizzate a due valori (*allow, not set*) o mostrando esplicitamente i tre valori

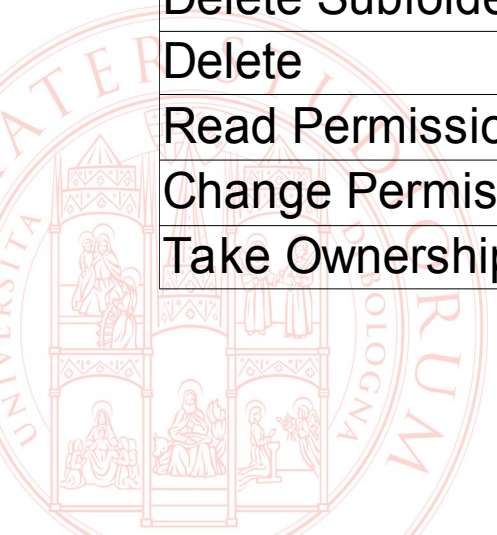
Autorizzazioni standard (elenco)

Abbreviation	Type	Description
R	Read	Provides the designated user or group the ability to read the file or the contents of the folder.
W	Write	Provides the designated user or group the ability to create or write files and folders.
RX	Read & Execute	Provides the designated user or group the ability to read file and folder attributes, view folder contents, and read files within the folder. If this permission is applied to a folder, files with inheritance set will inherit it (see the inheritance discussion).
L	List Folder Contents	Same as Read & Execute, but not inherited by files within a folder. However, newly created subfolders will inherit this permission.
M	Modify	Provides the ability to delete, write, read, and execute.
F	Full Control	Provides the ability to perform any action, including taking ownership and changing permissions. When applied to a folder, the user or group may delete subfolders and files within a folder.



Corrispondenza tra autorizzazioni standard e speciali

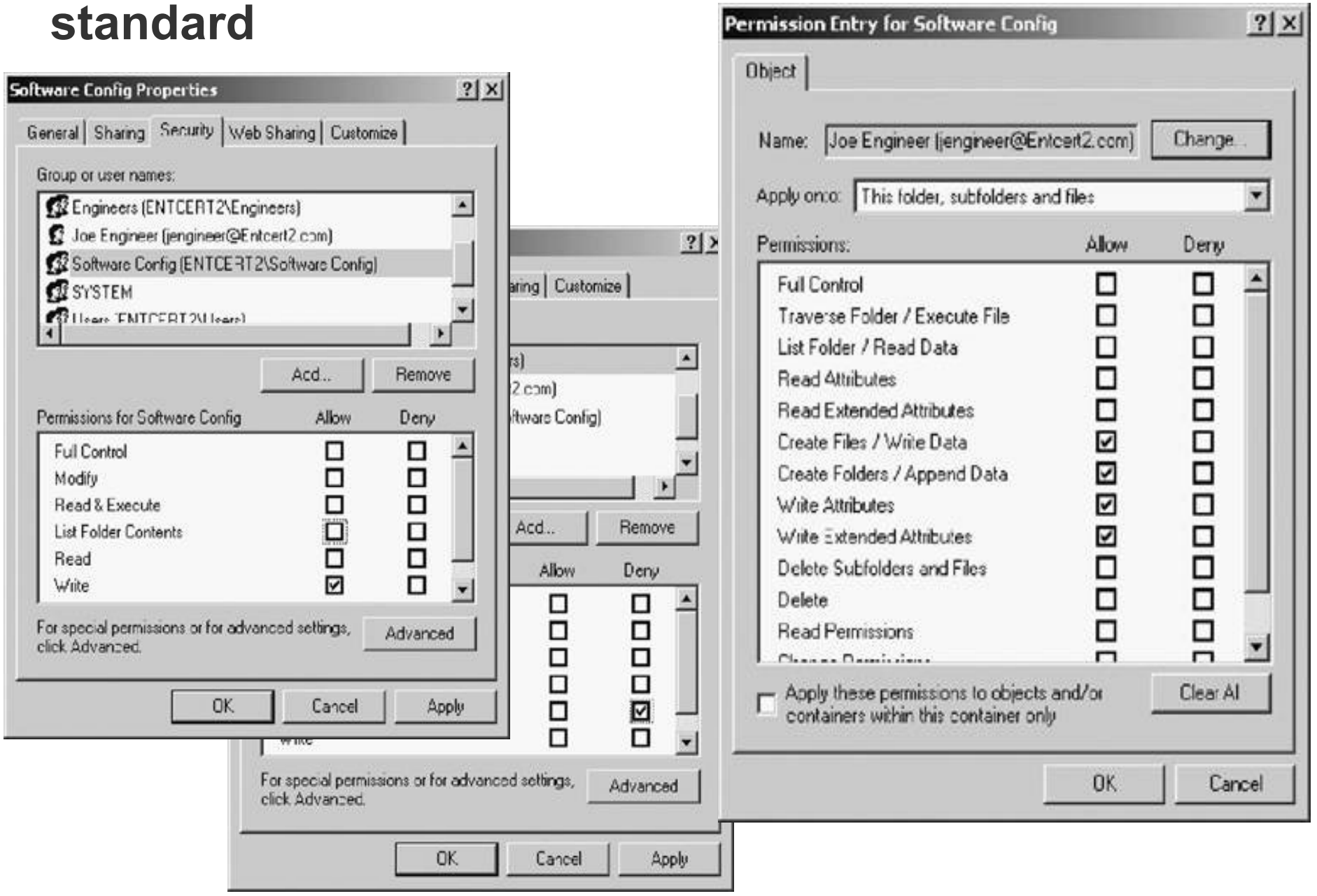
Types	Full Control	Modify	Read & Execute	List Folder Contents	Read	Write
Traverse Folder/Execute File	X	X	X	X		
List Folder/Read Data	X	X	X	X	X	
Read Attributes	X	X	X	X	X	
Read Extended Attributes	X	X	X	X	X	
Create Files/Write Data	X	X				X
Create Folders/Append Data	X	X				X
Write Attributes	X	X				X
Write Extended Attributes	X	X				X
Delete Subfolders and Files	X					
Delete	X	X				
Read Permissions	X	X	X	X	X	
Change Permissions	X					
Take Ownership	X					



ACL editing - esempi

standard

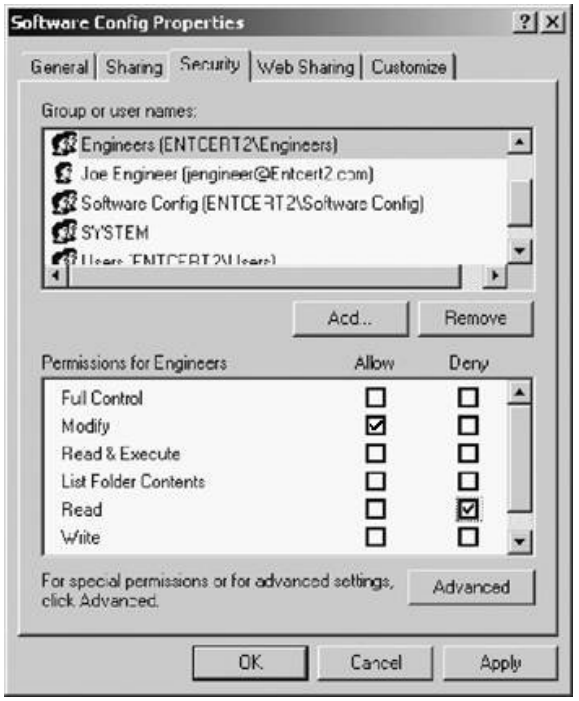
special



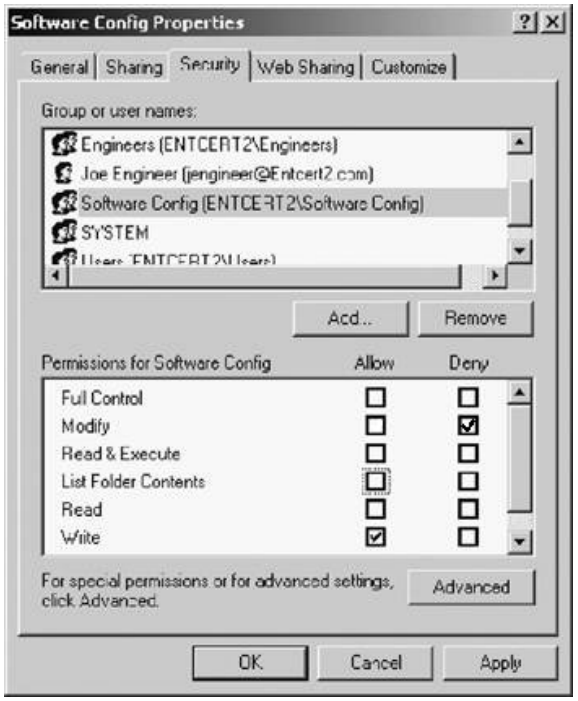
Composizione

- Come si può vedere, ogni permesso può essere impostato in tre modi
  - esplicitamente **allow**
  - esplicitamente **deny**
  - o **non impostato**
- Windows segue un modello default deny: ciò che non è esplicitamente consetito è proibito
  - quindi **non impostato** == **deny** ?
  - a cosa servono due modi diversi di proibire l'accesso?
- Un utente può appartenere a molti gruppi!
  - ogni gruppo può avere permessi distinti nell'ACL di una risorsa
  - il permesso complessivo dell'utente sarà la somma, bit a bit, di tutti i permessi ottenuti in quanto membro dei propri gruppi
- **non impostato** (sia allow che deny sono bit a zero) presente nei permessi di un gruppo consente che un **allow** ottenuto da un altro gruppo possa avere effetto: **non impostato == deny "debole"/scavalcabile**
- unt **deny** esplicito prevarrà sempre su di un eventuale **allow** ottenuto da un altro gruppo: **deny esplicito == deny "forte"/non scavalcabile**

Composizione



permessi dei membri del gruppo Engineers



permessi dei membri del gruppo Software Config

i membri del gruppo Software Config non potrebbero ottenere **modify** anche se appartenessero a un gruppo (come Engineers) che lo ha (strong deny)

i membri di Engineers non hanno il permesso **write** ma lo potrebbero ottenere se appartenessero a un gruppo (come Software config) che lo ha

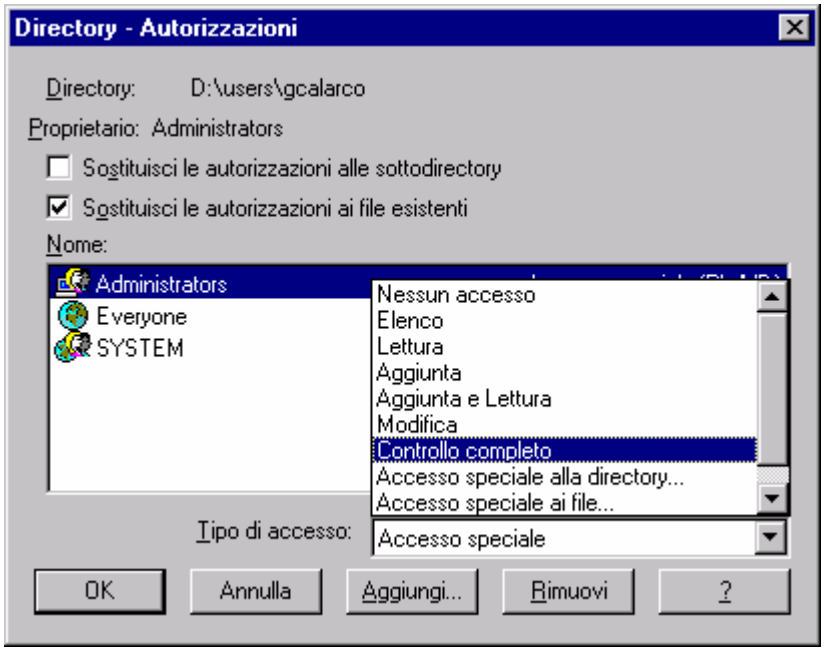
← non imp. resta non imp. == deny

← deny prevale su allow == deny

← deny esplicito == deny

← allow esplicito == allow

Esempio di manipolazione delle ACL (vecchia interfaccia semplificata)



Nelle nuove versioni di Windows, tutte le interfacce mostrano le colonne Allow e Deny, nelle vecchie esisteva una vista ancor più semplificata per scegliere

- una delle autorizzazioni standard (implicitamente equivalente a settare “Allow” su tutte le autorizzazioni speciali corrispondenti)
- oppure “Nessun accesso”, equivalente a deny su tutti i permessi

Autorizzazioni di accesso alle cartelle

- Sulle cartelle è possibile impostare una delle seguenti autorizzazioni standard:
  - Nessun accesso
  - Elenco
  - Lettura
  - Aggiunta
  - Aggiunta e Lettura
  - Modifica
  - Controllo completo
- Nota: I gruppi o gli utenti a cui è stata concessa l'autorizzazione ‘Controllo completo’ su una cartella sono in grado di eliminarne i file, indipendentemente dall'autorizzazione che li protegge.

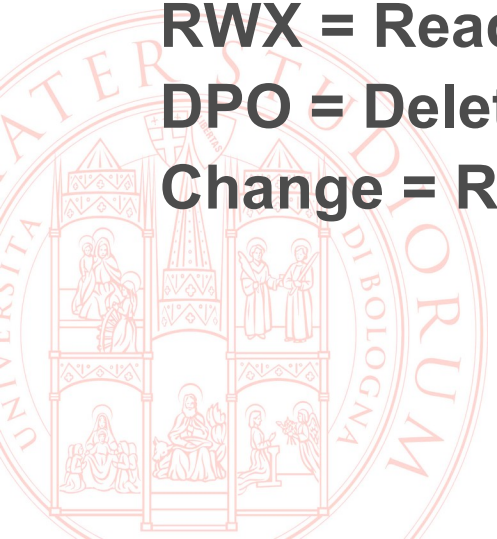
Autorizzazioni di accesso ai file

- Sui file è possibile impostare le seguenti autorizzazioni standard:
  - Nessun accesso
  - Lettura
  - Modifica
  - Controllo completo
- Impostando le autorizzazioni di accesso a un file sarà possibile specificare il tipo di accesso al file consentito a un gruppo o a un utente. Altrimenti, un file eredita le autorizzazioni proprie della cartella in cui è stato creato.
- Nota I gruppi o gli utenti cui si concede l'autorizzazione Controllo completo su una cartella possono eliminarne i file, indipendentemente dall'autorizzazione che li protegge.

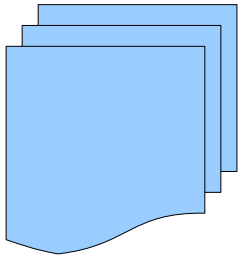
Esempio di composizione di permessi

Permessi di Michael	Permessi di Research	Permessi di Development	Permessi di Michael (effettivi)
Read	Read	-	Read
Write	-	Read	Change
Take Ownership	Read	Change	Take Ownership & Change
No Access	Read	Change	No Access
Change	No Access	Change	No Access

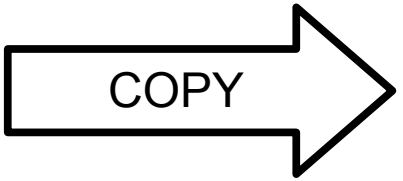
RWX = Read, Write, Execute
DPO = Delete, Permissions, Ownership
Change = RWXD

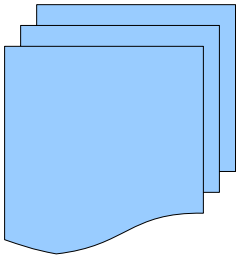


Permessi dopo un copy/move di file

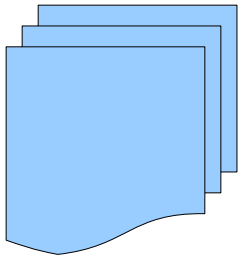


File1 = RWX

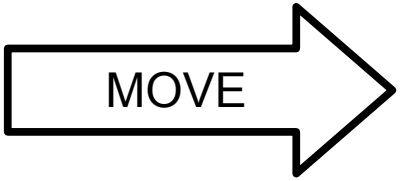


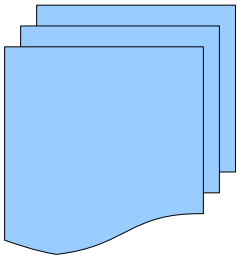


File1 = diritti del direttorio



File1 = RWX





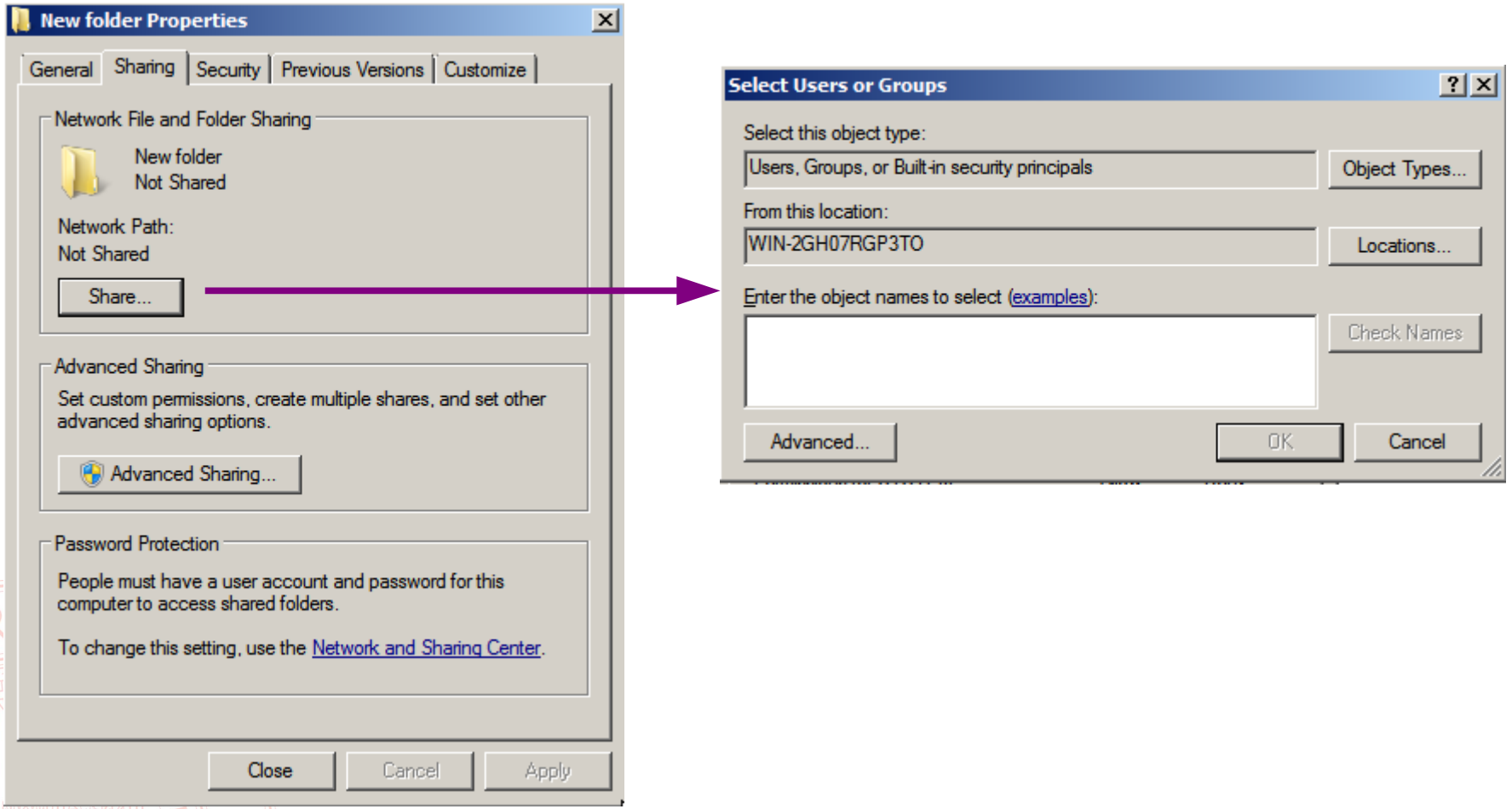
File1 = RWX

NOTA: MOVE verso una partizione diversa dalla sorgente = COPY (+delete)!

Condivisione di risorse

- Share = directory (folder) condivisa
- I diritti necessari per attivare le condivisioni sono concessi di default ai gruppi
  - Administrators
  - Server Operators (se in un dominio)
  - Power Users (se in un workgroup)
- Gli Users devono avere almeno il permesso List per fruire della directory condivisa

Condividere una cartella



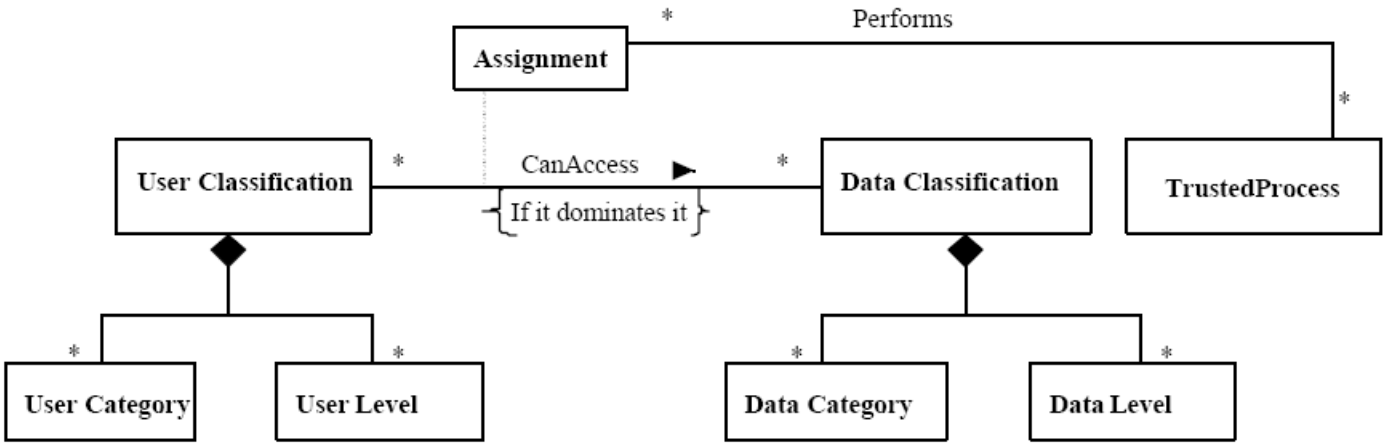
Permessi locali vs. Permessi sulla condivisione

	Permessi assegnati	Permessi di Michael
Permessi Share	Everyone: Read Michael: Change	Change (RWXD)
Permessi locali	Everyone: Read Michael: Read	Read (RX)
Permessi effettivi		Read (RX)

Le ACL di Share si comportano come quelle di NTFS in termini di composizione di permessi, però vengono applicate in serie una all'altra, per cui complessivamente l'autorizzazione effettiva è quella più restrittiva tra le due

Un breve cenno a MAC

- MANDATORY: le regole di controllo degli accessi sono dettate da un'autorità centrale e i soggetti non possono modificarne alcun dettaglio
- Ad ogni soggetto o risorsa viene assegnata una **classe di accesso** che specifica tipicamente
  - un livello di sicurezza all'interno di un insieme ordinato di valori, ad es {TopSecret ► Segreto ► Riservato ► Non classificato}
  - una categoria (**compartment**) all'interno di un insieme non ordinato, ad es. {armi, piani di battaglia, unità di combattimento, …} che riflette le aree funzionali del sistema



Come si usano le classi

- **le risorse** sono etichettate (**classification**) con un livello di sicurezza (**sensitivity**) che rappresenta la gravità delle conseguenze di una violazione delle policy che le riguarda
- **i soggetti** sono etichettati con un livello di sicurezza che rappresenta la loro affidabilità: **clearance**
- le categorie sono utilizzate per raffinare le politiche
- vengono stabilite relazioni di dominanza tra classi; detti
  - S un livello di sicurezza
  - C un insieme di categorie
  - $L_1 = \langle S_1, C_1 \rangle$
  - $L_2 = \langle S_2, C_2 \rangle$

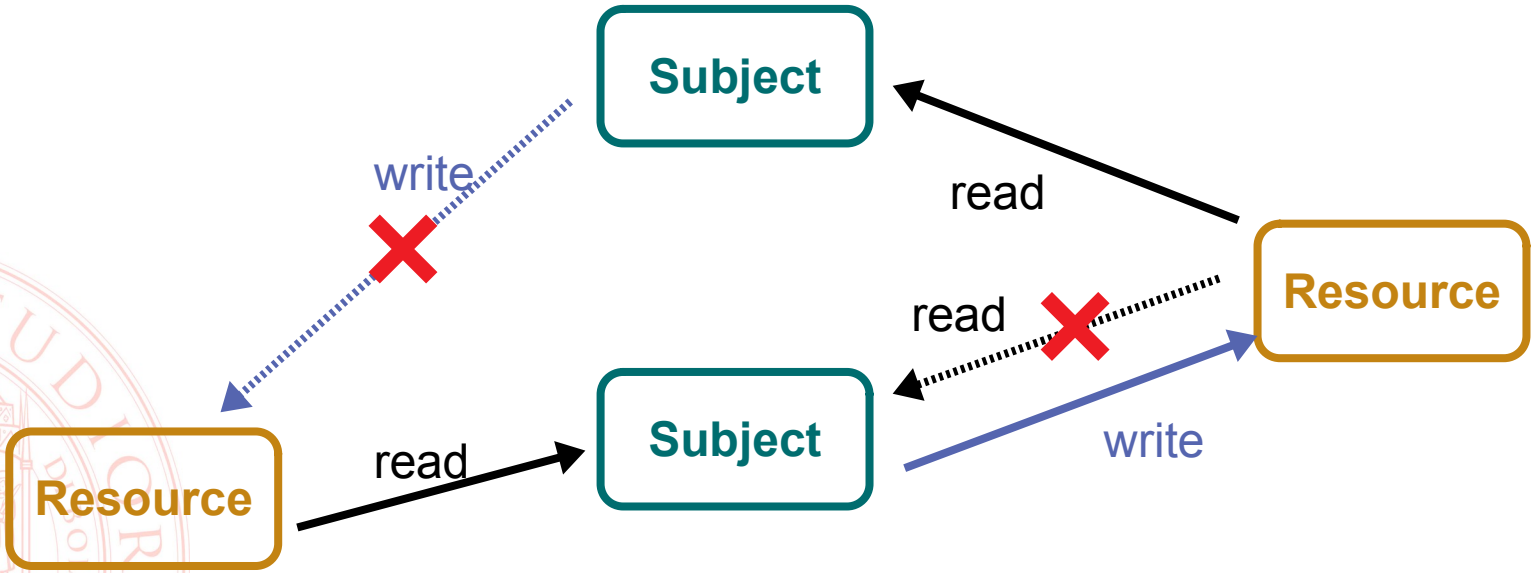
L_1 domina L_2 se e solo se $S_1 \geq S_2$ e $C_1 \supseteq C_2$

Come si usano le classi

- Le relazioni di dominanza vengono usate in modo diverso a seconda della proprietà di sicurezza da proteggere
  - riservatezza → Bell-LaPadula model
  - integrità → Biba model
- L'applicazione simultanea dei due modelli è possibile assegnando due classi di accesso a ogni soggetto e risorsa, una usata per controllare la riservatezza e l'altra per controllare l'integrità

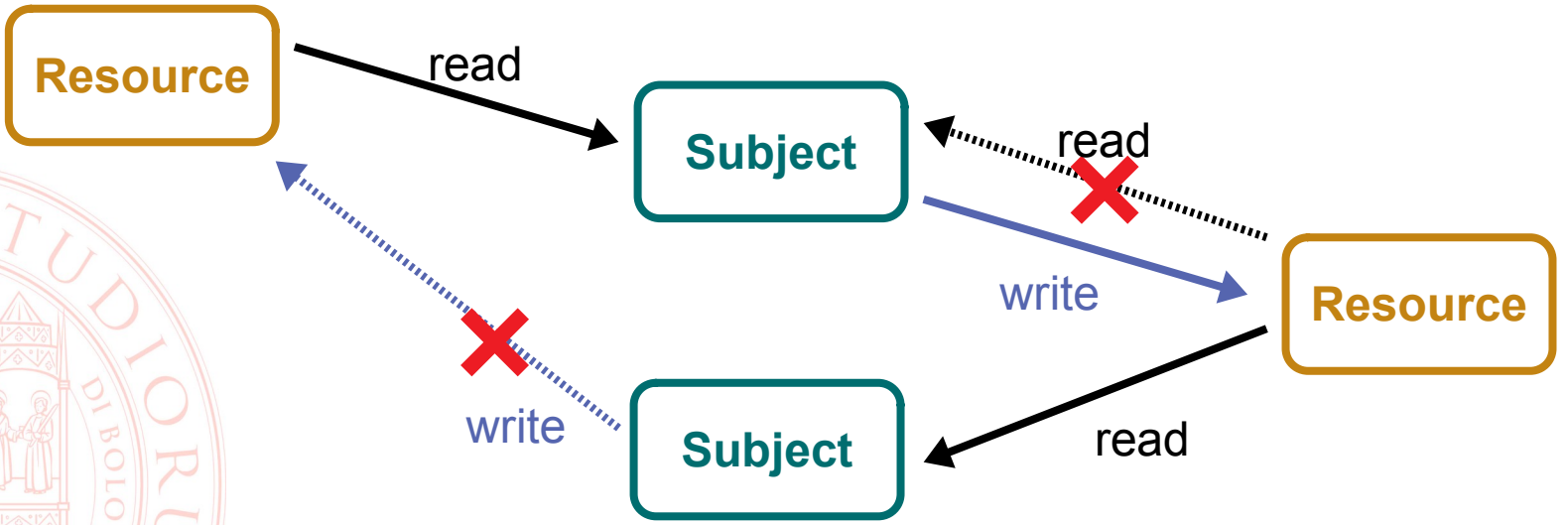
BLP (Bell-LaPadula)

- Due regole per proteggere la riservatezza
  - **NO-READ-UP**: un soggetto può leggere una risorsa solo se la sua classe di accesso domina la classe di accesso della risorsa (altrimenti leggerebbe una risorsa troppo sensibile per il suo livello)
  - **NO-WRITE-DOWN**: un soggetto può modificare una risorsa solo se la sua classe di accesso è dominata dalla classe di accesso della risorsa (altrimenti potrebbe far trapelare un segreto in luoghi accessibili a soggetti con minor clearance)



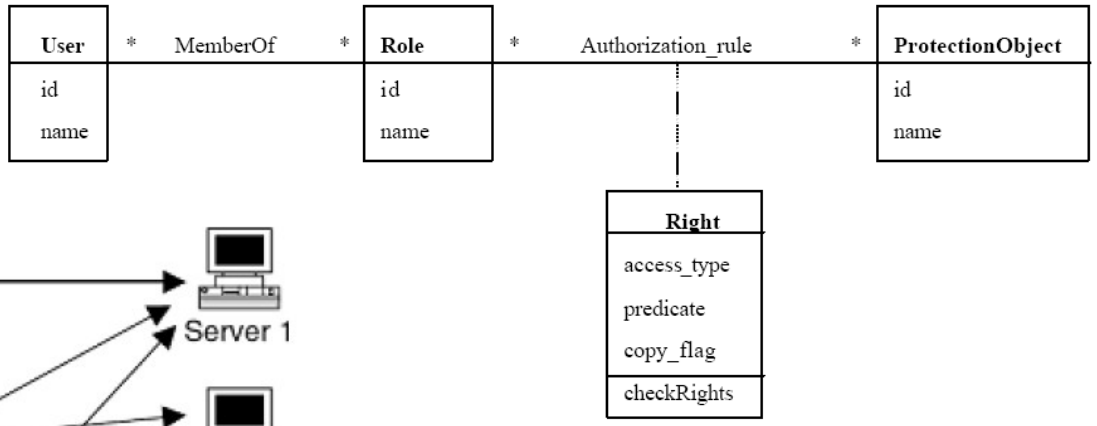
Biba

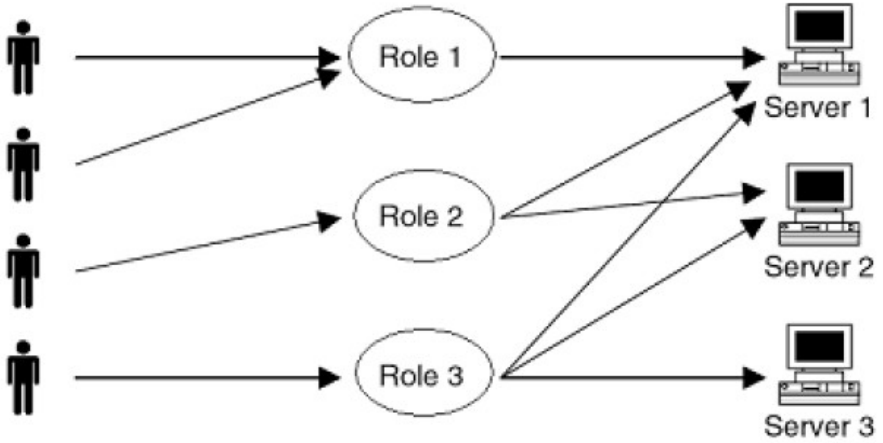
- Due regole per proteggere l'integrità:
  - **NO-READ-DOWN**: un soggetto può leggere una risorsa solo se la sua classe di accesso è dominata dalla classe di accesso della risorsa (altrimenti utilizzerebbe informazioni meno attendibili al proprio livello di fiducia più elevato)
  - **NO-WRITE-UP**: un soggetto può scrivere una risorsa solo se la sua classe di accesso domina la classe di accesso della risorsa (altrimenti modificherebbe una risorsa troppo sensibile per il suo livello)



Un brevissimo cenno a RBAC

- le autorizzazioni non sono concesse a utenti, ma a *ruoli*
- il ruolo ricoperto da un utente può cambiare dinamicamente
  - nel tempo
  - secondo il contesto





RBAC

- RBAC è un modello "policy neutral" che permette di esprimere tutti i principi fondamentali per la sicurezza:
  - minimo privilegio
  - separazione delle responsabilità
  - astrazione (es. usando generici "debiti" e "crediti" al posto di permessi specifici delle risorse come “possibilità di lettura/scrittura”)
- Vantaggio: le autorizzazioni cambiano poco o per nulla, se correttamente modellate
  - il ruolo dell’amministratore della sicurezza diventa essenzialmente quello di assegnare il ruolo appropriato ai soggetti
- Esiste un modello standard

(ANSI/INCITS 359-2004)

con livelli di funzionalità

definiti in modo incrementale

per adattare semplicemente

la sua implementazione in

contesti di sicurezza differenziati